Face Detection Using Deep Learning

Yu Shen, Kuan-Wei Chen, Yizhou Hao, Min Hsuan Wu
Department of Electrical and Computer Engineering, University of California, San Diego

Abstract

Face detection is becoming a very popular technology now. It has been implemented and widely used in different industries and products. Face detection is definitely going to be an indispensable part in the intelligent society in the near future. This poster here presents a project that implements face detection by using deep learning. The specific method used in this project is Multi-task Cascaded Convolutional Neural Network(MTCNN). And the database used in this project is FDDB dataset. The project uses modified method from current existing method.

Methods

The network we used for face detection is called Multi-task Cascaded Convolutional Neural Network(MTCNN). This algorithm contains three stages of computations for face detection. The pipeline of the algorithm can be seen in figure one and two. First of all, the image will be resized and be built in to a image pyramid. After that, in the first stage, which is called Proposal net, the face classification, bounding box regression and facial landmark localization will be computed. These data is the input to the second stage, which is called Refine net. In this stage, convolutional layer combined with fully connected layer will improve the result from previous stage. Then, the refined result becomes the input to the third stage, the Output net. This stage is similar to previous one, but the restriction of five key facial landmarks will do a better suppression to those boxes that don't contains faces.

Figure 1. MTCNN Structure

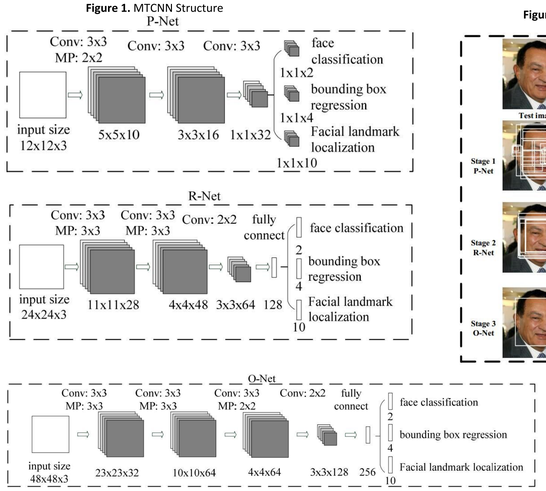
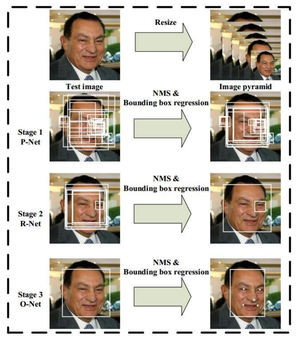


Figure 2. Stages of MTCNN



Data

The data used is FDDB dataset. It contains 5171 faces in a set of 2845 (both grayscale and colored) images with coordinate labels an ellipse around each human face with radius of both axis, center of the ellipse, and the angle of the ellipse. Each image may contain multiple faces.

Figure 3a. Sample data contains multiple faces in a single image.

Figure 3b. Sample data with ground truth label.

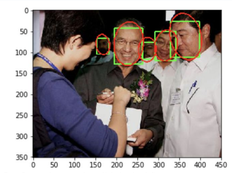


Figure 3c. Sample result, all labeled faces are correctly detected. Red ellipse is ground truth label, green box is detection result.

Results

Folder	False positive	Correct detection	Total faces	Accuracy	True Positive Rate
1	23	480	515	0.932	0.954
2	18	485	519	0.934	0.964
3	25	483	517	0.934	0.951
4	18	488	517	0.944	0.964
5	17	486	514	0.946	0.966
6	21	484	518	0.934	0.958
7	30	494	518	0.954	0.943
8	24	468	518	0.903	0.951
9	15	483	514	0.940	0.970
10	24	495	521	0.950	0.954
All	215	4846	5171	0.937	0.958

Accuracy is the number of correct detection divided by the number of true faces. True positive rate is the number of correct detection divided by the total number of detections.

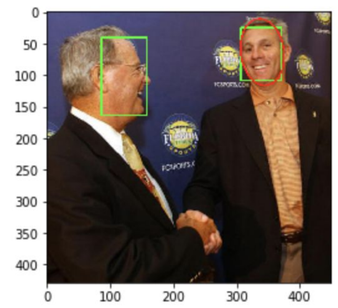


Figure 4. Sample false positive result. Green border is the detection results, red ellipse is the ground truth label. Note that the face on the left is not labeled as neither of the two eyes were visible

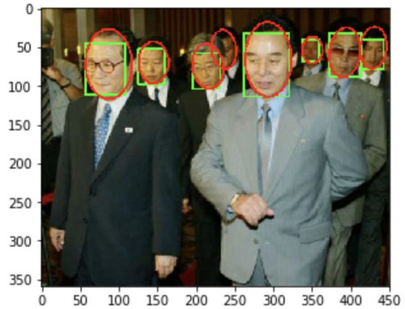


Figure 5. Sample false negative result. Resolution, occlusion, and pose all affect the label of whether it counts as a "face". In this case since the occluded face still has 1 visible eye, it was labeled as a face but was not detected.

Comparison

Before MTCNN, the most popular method is Viola-Jones method from openCV. It is a Haar Feature-based Cascade Classifier for Object Detection method, which to find most frequent features such as edge, line, and center-around features. However, to determing how many classfiers need to be used in Viola-Jones, it has to have a tradeoff between false positive rate and detection rate. Hence, it will not be as fast and accurate as MTCNN method.

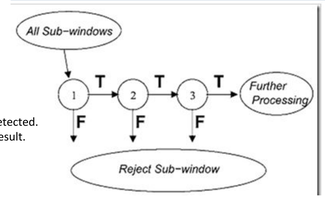


Figure 6. Viola-Jones method process

Conclusions

By using FDDB dataset for evaluation, our implementation of MTCNN get average 93.7% of accuracy. We think the result is pretty good, but it can still be improved. We look into the evaluation to find the miss-detections cases. As an example, like figure 4, we got some false positive cases. The face was not label since neither of the two eyes were visible. In figure 5, one face was not detected since half of the face is not visible. It shows the dataset's labels are not perfect. The dataset's labels are more focusing on eyes of the face. However, MTCNN does not only depend on eyes. The differences lead to the negative results, which causes our accuracy to go low. For the future work, we can find larger datasets to improve the performance of our detector. By doing more evaluations, we can find the weakness of our implementation, and we can improve it step by step.

References

Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks

https://arxiv.org/abs/1604.02878

Modern Face Detection based on Deep Learning using Python and Mxnet
https://medium.com/wassa/modern-face-detection-based-on-deep-learning-using-python-and-mxnet-5e6377f22674

FDDB Dataset
http://vis-www.cs.umass.edu/fddb/index.html